

AD-A118 126

GEORGE WASHINGTON UNIV WASHINGTON DC INST FOR MANAGE--ETC F/G 12/1
CONCAVE MINIMIZATION VIA COLLAPSING POLYTOPES.(U)
DEC 80 J E FALK, K L HOFFMAN

DAA629-79-C-0062

NL

UNCLASSIFIED

SERIAL-T-438

1 0 1

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

0 1 0 1 0

END

DATE

FILED

9 82

DTIC

AD A118126

DTIC FILE COPY

THE
GEORGE
WASHINGTON
UNIVERSITY

STUDENTS FACULTY STUDY R
ESEARCH DEVELOPMENT FUT
URE CAREER CREATIVITY CO
MMUNITY LEADERSHIP TECH
NOLOGY FRONTIER DESIGN
ENGINEERING APP ENO
GEORGE WASHINGTON UNIV

DTIC
ELECTE
AUG 10 1982
S H D



SCHOOL OF ENGINEERING
AND APPLIED SCIENCE

THIS DOCUMENT HAS BEEN APPROVED FOR RELEASE

12

CONCAVE MINIMIZATION VIA COLLAPSING POLYTOPES

by.

James E. Falk
Karla L. Hoffman

Serial T-438
16 December 1980

The George Washington University
School of Engineering and Applied Science
Institute for Management Science and Engineering

DTIC
ELECTE
S AUG 10 1982
H

Army Research Office
Contract DAAG 29-79-C-0062

Office of Naval Research
Contract N00014-75-C-0611

National Bureau of Standards
Operations Research Division

This document has been approved for public
sale and release; its distribution is unlimited.

THE GEORGE WASHINGTON UNIVERSITY
School of Engineering and Applied Science
Institute for Management Science and Engineering

Abstract
of
Serial T-438
16 December 1980

CONCAVE MINIMIZATION VIA COLLAPSING POLYTOPES

by

James E. Falk†
Karla L. Hoffman*

The global minimization of a concave function over a (bounded) polytope is accomplished by successively minimizing the function over polytopes containing the feasible region, and collapsing to the feasible region. The initial containing polytope is a simplex, and, at the k th iteration, the most promising vertex of the current containing polytope is chosen to refine the approximation. A tree whose ultimate terminal nodes coincide with the vertices of the feasible region is generated, and accounts for the vertices of the containing polytopes.

†Dr. Falk's research was supported in part by the Army Research Office under Contract DAAG 29-79-C-0062, the Office of Naval Research under Contract N00014-75-C-0611, and the Operations Research Division of the National Bureau of Standards.

*National Bureau of Standards.

NONE

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER T-438	2. GOVT ACCESSION NO. AD-1118 126	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) CONCAVE MINIMIZATION VIA COLLAPSING POLYTOPES		5. TYPE OF REPORT & PERIOD COVERED SCIENTIFIC
7. AUTHOR(s) JAMES E. FALK KARLA L. HOFFMAN		6. PERFORMING ORG. REPORT NUMBER T-438
9. PERFORMING ORGANIZATION NAME AND ADDRESS GEORGE WASHINGTON UNIVERSITY INSTITUTE FOR MANAGEMENT SCIENCE & ENGINEERING WASHINGTON, DC 20052		8. CONTRACT OR GRANT NUMBER(s) DAAG 29-79-C-0062 N00014-75-C-0729
11. CONTROLLING OFFICE NAME AND ADDRESS OFFICE OF NAVAL RESEARCH CODE 434 ARLINGTON, VIRGINIA 22217		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		12. REPORT DATE 16 December 1980
		13. NUMBER OF PAGES 30
		15. SECURITY CLASS. (of this report) NONE
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) APPROVED FOR PUBLIC SALE AND RELEASE; DISTRIBUTION UNLIMITED.		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) CONCAVE MINIMIZATION BRANCH AND BOUND GLOBAL OPTIMIZATION		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) The global minimization of a concave function over a (bounded) polytope is accomplished by successively minimizing the function over polytopes containing the feasible region, and collapsing to the feasible region. The initial containing polytope is a simplex, and, at the kth iteration, the most promising vertex of the current containing polytope is chosen to refine the approximation. A tree whose ultimate terminal nodes coincide with the vertices of the feasible region is generated, and accounts for the vertices of the containing polytopes.		

DD FORM 1473
1 JAN 73

EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6001

NONE

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

THE GEORGE WASHINGTON UNIVERSITY
School of Engineering and Applied Science
Institute for Management Science and Engineering

CONCAVE MINIMIZATION VIA COLLAPSING POLYTOPES

by

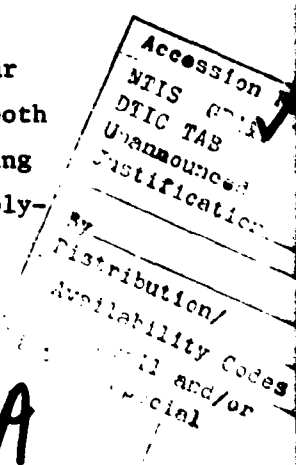
James E. Falk
Karla L. Hoffman

1. Introduction

In [1], the authors introduced the Successive Underestimation Method (SUM) for the minimization of a concave function over a (bounded) polytope. This method has the features

- (a) it is a finite procedure,
- (b) it does not involve cuts of the feasible region,
- (c) it does not require any accounting scheme to keep track of subproblems,
- (d) it generates a sequence of linear programs,
- (e) it employs only pivoting operations, and
- (f) it is guaranteed to terminate at a global solution.

The method which we are introducing in this paper has similar characteristics and, indeed, has a similar interpretation in that both methods seek a global minimizer by successively collapsing containing polytopes around the feasible region. The method by which these polytopes are collapsed is, however, very different.



In SUM, at iteration k , one has a polytope S_k containing the feasible region S . This polytope S_k has both a representation in terms of linear inequalities (e.g., $A_i x \leq b_i$, $i \in I_k$), and also a representation in terms of its vertices (i.e., $\{w^j: j \in J_k\}$ is the set of vertices of S_k). Both of these representations are known at iteration k . To proceed, one selects the most promising vertex w^{j_k} of S_k , and identifies a halfspace $A_{i_k} x \leq b_{i_k}$ which does not contain w^{j_k} . This halfspace is then added to the halfspaces defining S_k to create S_{k+1} . One must then generate the new vertices defining S_{k+1} , and this is the most expensive part of the algorithm.

By contrast, in our new method, we will only carry along the vertex representation of S_k . As before, the most promising vertex w^{j_k} of S_k is selected, but now we generate S_{k+1} by listing the neighbors of a point v^{j_k} associated with w^{j_k} in a space of dimension one greater than the dimension of w^{j_k} . The new vertices defining S_{k+1} will be identified by these neighbors (and there will be at most $n+1$ of them). Thus the amount of work required to pass from iteration k to $k+1$ is greatly reduced.

A pleasant by-product of the proposed method is that redundant constraints do not affect the computation (as they do in SUM). This fact was established by Mattheiss [2], who proposed a method similar to the scheme we will present, but for the generation of *all* vertices of a polytope. Indeed, our method may be viewed as our original algorithm, SUM, modified along the lines proposed by Mattheiss for a different problem.

It is intended that a reader of this paper need not be intimately aware of the results of [1] and [2], i.e., we intend this paper to stand

alone. However, we will not repeat the proofs of those results that we need from those references.

In Section 2, we will list the assumptions and notation that we use later on. The algorithm will be presented here, along with a pair of illustrative examples. Section 3 contains the necessary proofs, and computational results are contained in Section 4.

2. The Method of Collapsing Polytopes

The problem that we address has the form

$$\left. \begin{array}{l} \text{minimize } f(x) \\ \text{subject to } Ax \leq b \end{array} \right\} \text{ Problem P}$$

where f is a concave function defined over R^n , and A is an $m \times n$ matrix with $m > n$. We shall assume that

- (a) the set $S = \{x: Ax \leq b\}$ is compact,
- (b) S has a nonempty interior,
- (c) $x > 0$ for all $x \in S$,
- (d) any basic solution of the system $C = \{v = (x, y): Ax + ay \leq b\}$
 $(a = (||A_1||, \dots, ||A_m||))$ is nondegenerate, and
- (e) the solution of the linear program

$$\left. \begin{array}{l} \text{maximize } y \\ \text{subject to } Ax + ay \leq b \end{array} \right\} \text{ Problem CP}$$

is unique.

Assumption (a) guarantees the existence of a solution to problem P, but it is also necessary to invoke it according to the nature of the algorithm that we are proposing. Assumption (b) is also essential, as our method needs to identify a center of the feasible region. Via Assumption (e), that center will be unique, and this will significantly simplify our description of the method. Likewise, Assumptions (c) and

(d) are not essential, but rather convenient to invoke for expository purposes. In particular, under (d) any vertex of the set C is determined by precisely $n+1$ hyperplanes of the form $A_i x + ||A_i|| y = b_i$.

Note that the feasible region $S = \{x: Ax \leq b\}$ can be interpreted as that n -face of the set $C = \{(x,y): Ax + ay \leq b\}$ for which $y = 0$. In particular, the vertices of S are those vertices of C for which $y = 0$.

If $x \in S$, the Euclidean distance between x and a hyperplane $A_i x = b_i$ is

$$\frac{b_i - A_i x}{||A_i||}.$$

Thus the quantity

$$y = \min_{1 \leq i \leq m} \left\{ \frac{b_i - A_i x}{||A_i||} \right\}$$

measures the distance from x to the nearest bounding hyperplane of S . It follows that the solution (x^0, y^0, s^0) of the problem CP,

$$\begin{aligned} &\text{maximize } y \\ &\text{subject to } Ax + ay + s = b \\ &\quad x, y, s \geq 0 \end{aligned}$$

yields a point $x^0 \in S$ which is the center of the largest sphere contained in S . The value y^0 is the radius of this sphere.

Because of Assumption (d), the solution (x^0, y^0, s^0) of CP is nondegenerate. Since there are m equality constraints defining CP, it follows that exactly m of the components of (x^0, y^0, s^0) are positive. Because of (c), $x^0 > 0$, and because of (b), $y^0 > 0$. Thus the $n+1$ nonbasic components of (x^0, y^0, s^0) are among the components of s^0 .

The first step in our algorithm will be to identify the $n+1$ neighbors (x^i, y^i, s^i) of (x^0, y^0, s^0) . This can be done algorithmically by representing (x^0, y^0, s^0) in tableau form and pivoting in those $n+1$ components of s^0 which are zero (see Figure 1).

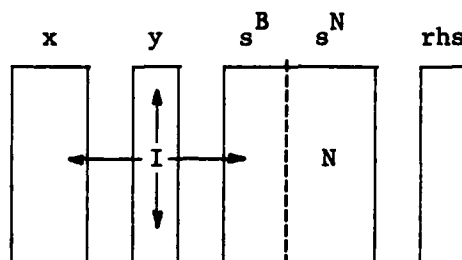


Figure 1.--The last tableau of CP.

In order to generate the first enclosing polytope S_0 of S , we need to extend the rays beginning at (x^0, y^0) and passing through the neighbors (x^i, y^i) until these rays pierce the hyperplane $y = 0$. It will be shown in Section 3 that this is possible for each of the neighbors (x^i, y^i) of (x^0, y^0) . Algorithmically, this is accomplished by simply pivoting on each element in that row which corresponds to the (basic) variable y in the tableau representing (x^0, y^0, s^0) . Such pivots result in points e^i which may or may not be feasible to S . We will, however, show in Section 3 that S is a subset of the convex hull of these points.

At stage k , we will have a tree whose terminal nodes $v^t = (x^t, y^t)$ will be vertices of C , where $C = \{v = (x, y) : Ax + ay \leq b\}$. (This is consistent with the tree built by Matheiss [2] to generate all vertices of a polytope.) Associated with each of these nodes will be a value f_t corresponding to the objective function f evaluated at a point associated with v^t . We will choose that vertex v^t which has

an associated f_t value that is minimal over all such terminal vertices.

Given that tableau which represents v^t , we see from Assumption (d) that there are precisely $n+1$ neighbors of v^t on C . However, we only focus on those neighbors $v^{t,i} = (x^{t,i}, y^{t,i})$ such that

(a) $y^{t,i} < y^t$, and

(b) $v^{t,i}$ is not a member of the tree associated with stage k .

Neighbors satisfying (a) can be identified from the tableau representing v^t . The row of that tableau corresponding to the variable y represents the equation

$$y + \sum_{j \text{ nonbasic}} a_{ij} s_j = y^t.$$

Thus, the only columns j which are eligible for consideration under (a) have $a_{ij} > 0$. Those columns ineligible under (b) must be identified by a tree search, which can be facilitated by the values y^t and $y^{t,i}$.

As before, we extend the rays emanating from v^t through the points $v^{t,i}$ until they pierce the plane $y=0$. This is accomplished by pivoting on the entry a_{ij} identified above. The result is a set of (at most $n+1$) points $e^{t,i}$. We compute the values $f_{t,i} = f(e^{t,i})$, and associate these values with the neighbors $v^{t,i}$. (The vertex $v^{t,i}$ obtained by doing the usual ratio test to determine the leaving variable.)

We will show in Section 3 that S is a subset of the convex envelope of the points $\{e^t\}$ from stage k , but *excluding* the particular

point e^t whose associated v^t was used to extend the tree, but including the newly generated points $e^{t,i}$.

We may summarize the algorithm as follows.

Initialize: Solve problem CP and obtain solution $v^0 = (x^0, y^0)$. Set $f_0 = \infty$,
 $T_0 = \{(v^0, f_0)\}$.

Select Branching Node: With T_k given, select a terminal node v^t whose associated value f_t is minimal over all such terminal nodes. If the associated $y^t = 0$, stop with global solution $x^* = x^t$ and $f^* = f_t$.

Step: Generate all neighbors $v^{t,i}$ of v^t such that
 (a) $y^{t,i} < y^t$, and
 (b) $v^{t,i} \notin T_k$.

For each such $v^{t,i}$, compute

$$e^{t,i} = x^t + \left(\frac{y^t}{y^t - y^{t,i}} \right) (x^{t,i} - x^t)$$

and $f_{t,i} = f(e^{t,i})$. The tree T_{k+1} is the tree T_k with new nodes $v^{t,i}$ satisfying (a) and (b) above, and with links joining v^t to these new $v^{t,i}$.

Example 1:

The problem we wish to solve is

$$\begin{aligned}
 &\text{minimize} && -(x_1-2)^2 - (x_2-2)^2 \\
 &\text{subject to} && x_1 + x_2 \geq 1 \\
 & && x_1 - 2x_2 \leq 1 \\
 & && 2x_1 - x_2 \leq 5 \\
 & && 3x_1 + 5x_2 \leq 27 \\
 & && -6x_1 + 10x_2 \leq 30 \\
 & && x_1 \geq 0.
 \end{aligned}$$

This example was constructed so that there are four global minimizers, and all vertices of S need to be generated. The final tableau of problem CP is given in Table 1. The circled entries indicate pivot elements which yield the three neighbors of $v^0 = (x^0, y^0) = ((1.799, 2.201), 1.611)$ on C . The starred entries indicate pivots which yield the extended neighbors of v^0 . The fact that one entry is both circled and starred indicates that a pivot here will yield a neighbor (x^1, y^1) where x^1 is actually feasible to S .

The complete solution tree is given in Figure 2. A global solution was found at node 4, but not verified as such until the alternate solution at nodes 8 and 9 were generated.

The geometrical history of the collapsing polytopes is depicted in Figure 3.

Example 2:

This problem is based on a problem derived by Zwart [4] as a counterexample to Tiu's original method [3] for minimizing concave functions over polyhedra. The problem is

TABLE 1
FINAL TABLEAU OF PROBLEM CP

x_1	x_2	y	s_1	s_2	s_3	s_4	s_5	s_6	b
1	0	0	0	-0.481	0.646	0	-0.032	0	1.799
0	1	0	0	-0.519	0.354	0	0.032	0	2.201
0	0	1	0	0.198*	0.028*	0	0.042*	0	1.611
0	0	0	1	-1.280	0.960	0	-0.060	0	0.722
0	0	0	0	-0.679	0.617	0	-0.074	1	0.188
0	0	0	0	2.885	-3.874	1	-0.310	0	1.205
0	0	0	0	0.198	0.028	0	0.042	0	1.611

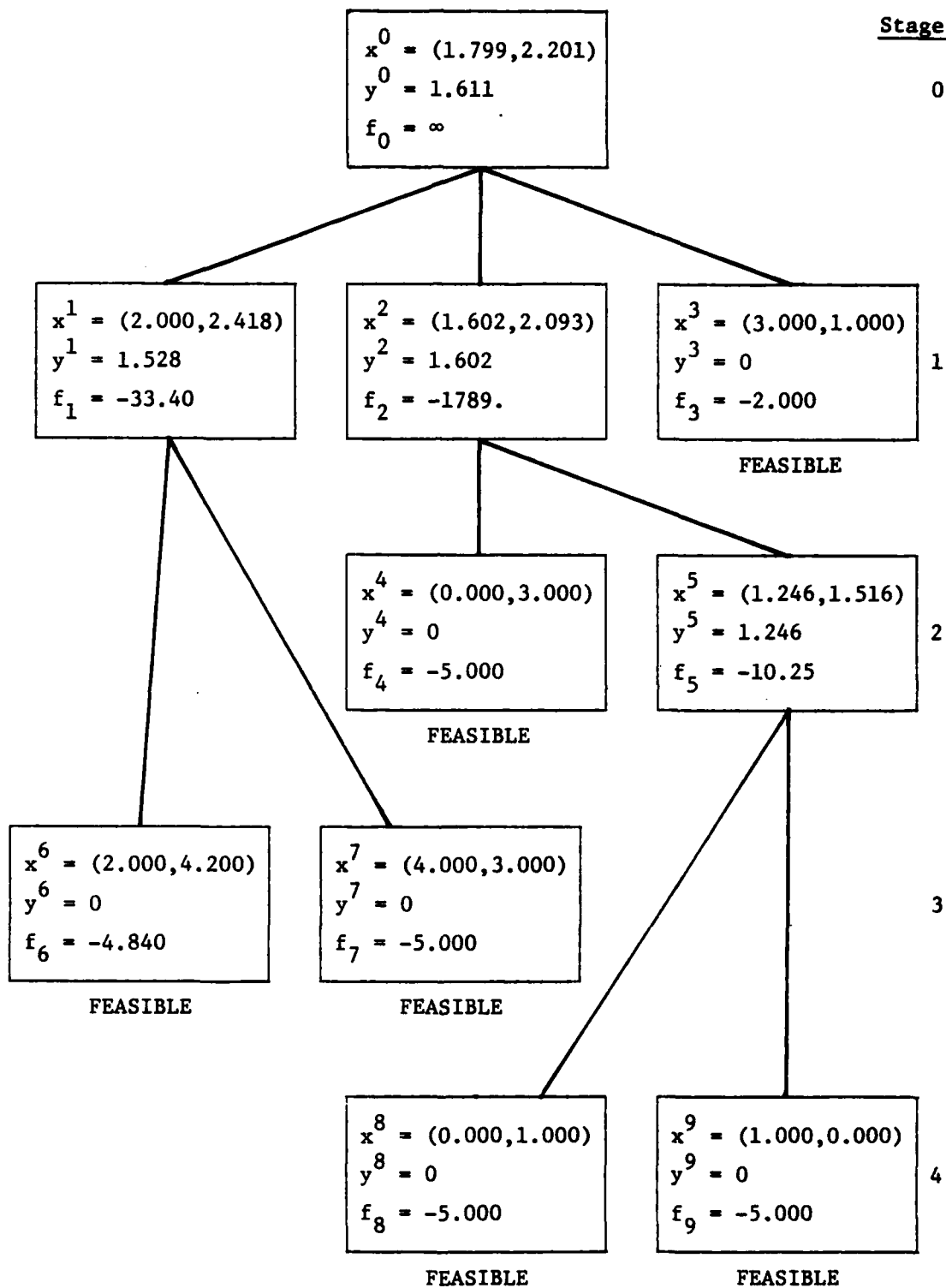


Figure 2.--The tree for Example 1.

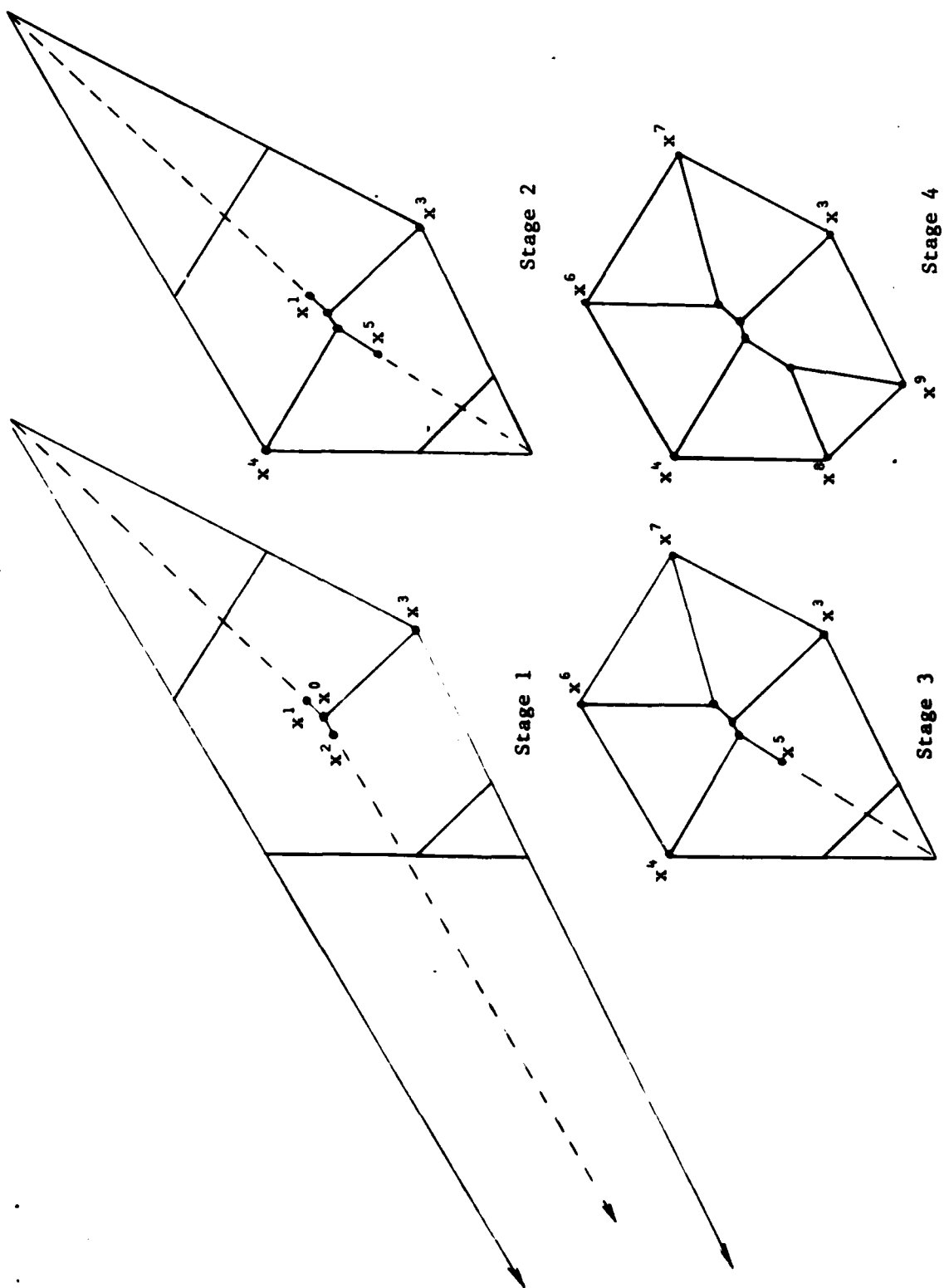


Figure 3.--Collapsing polytopes.

$$\begin{aligned}
& \text{minimize} \quad -(x_1-1)^2 - x_2^2 - (x_3-1)^2 \\
& \text{subject to} \quad 4x_1 - 5x_2 + 4x_3 \geq 4 \\
& \quad \quad \quad 6x_1 - x_2 - x_3 \geq 4.1 \\
& \quad \quad \quad x_1 + x_2 - x_3 \leq 1 \\
& \quad \quad \quad 12x_1 + 5x_2 + 12x_3 \leq 34.8 \\
& \quad \quad \quad 12x_1 + 12x_2 + 7x_3 \leq 29.1 \\
& \quad \quad \quad x_1, x_2, x_3 \geq 0.
\end{aligned}$$

The algorithm required three stages to uncover and verify the global solution at $x^* = (1,0,0)$. An interesting feature of this example occurs in the first stage, where problem 3 is selected for branching. As expected, (x^3, y^3) has four neighbors, one of which is (x^0, y^0) . However, there is also a neighbor that is already on the list $((x^4, y^4))$, and hence should not be added again, in spite of the fact that the y value associated with this neighbor (0.147) is smaller than the current $y^3 = 0.214$ value. Thus a list search is necessary.

The branch and bound tree is depicted in Figure 4, and the geometrical history of the method is depicted in Figure 5.

3. Proof of Convergence

In order to prove convergence of this method, it suffices to show that (a) the initial polytope generated by the convex hull of the extended neighbors of v^0 contains the feasible region S , and (b) if the polytope associated with tree T_k contains S , then the polytope associated with T_{k+1} contains S . Since we are growing trees whose nodes are in one-to-one correspondence with vertices of C , the method must be finite.

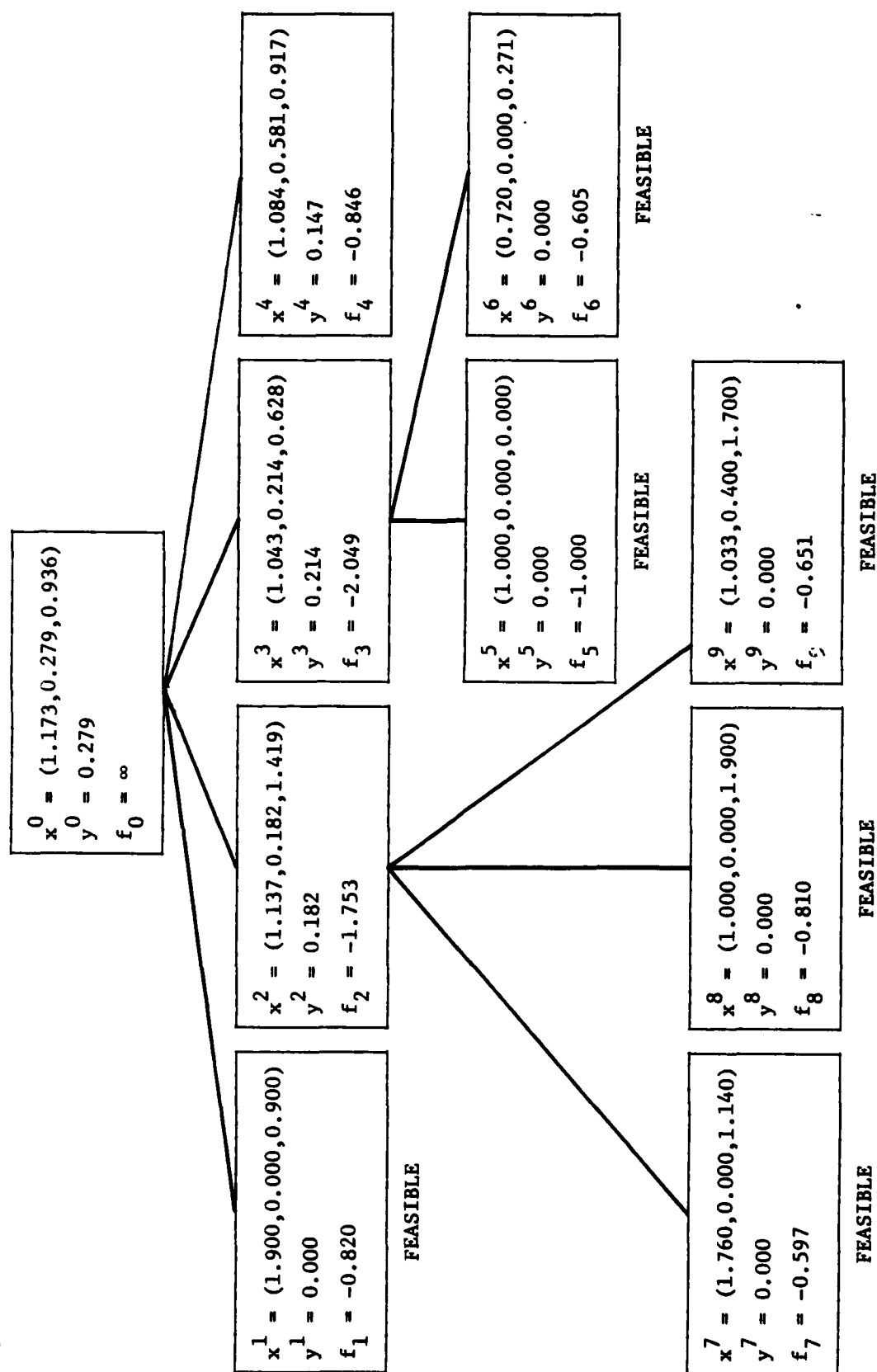


Figure 4.--Branch and bound tree for Example 2.

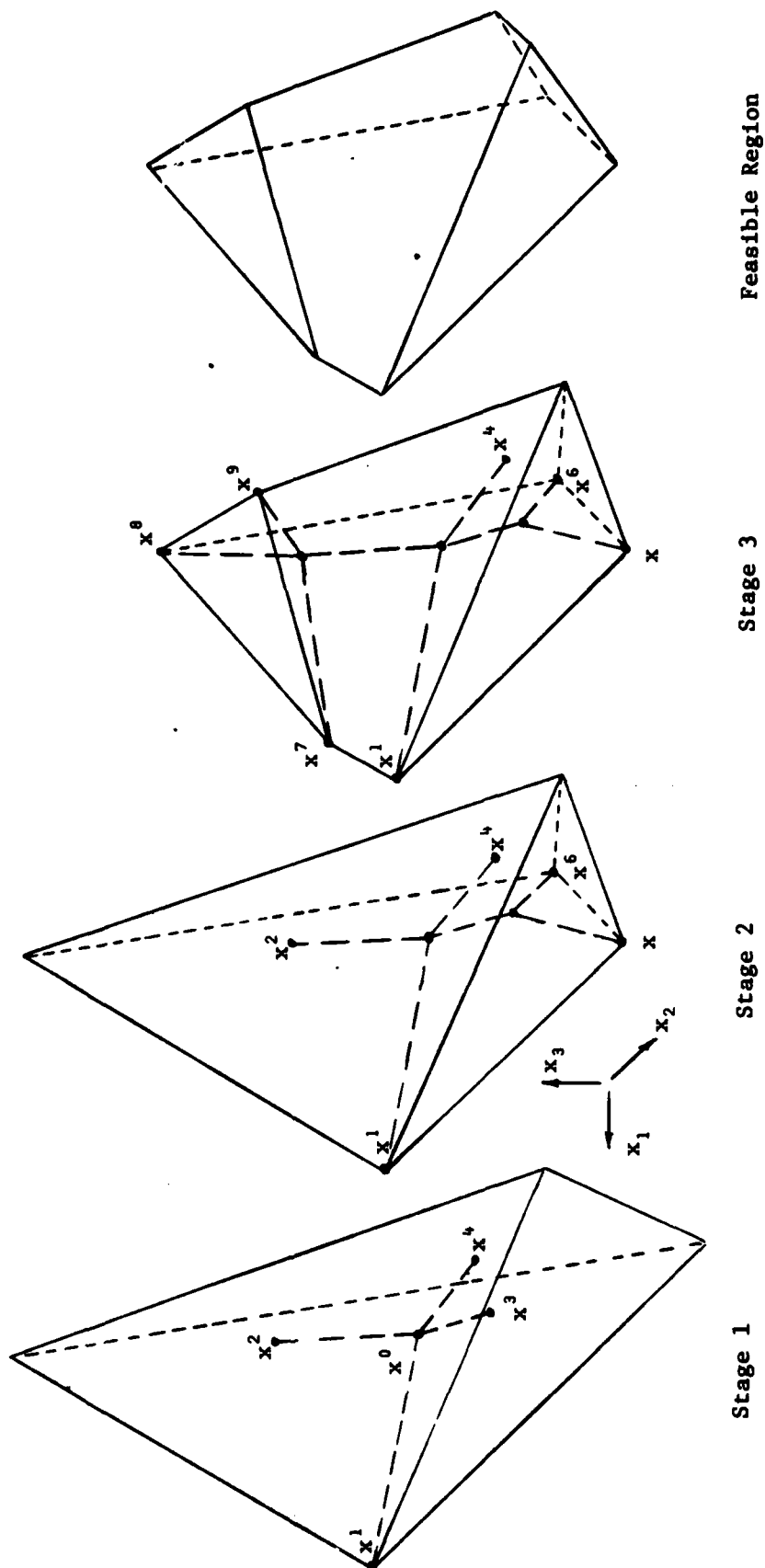


Figure 5.--Second example.

The vertices of C form a directed graph $G = \{v, E\}$ with edges $(v^s, v^t) \in E$ if and only if $v^s = (x^s, y^s)$ and $v^t = (x^t, y^t)$ are neighboring vertices on C , and if $y^s > y^t$ (i.e., x^t is closer to the boundary of S than is x^s). The algorithm that we are describing will specify a sequence T_0, T_1, \dots, T_p of subtrees of G such that

- (a) $T_0 = \{(x^0, y^0)\}$ where x^0 solves problem CP,
- (b) T_{k+1} is defined from T_k by selecting a terminal vertex $v_k^{i_k}$ of T_k , generating all neighbors $v_k^{i_k, j}$ of $v_k^{i_k}$ which are not already in T_k , and whose associated distance values $y_k^{i_k, j}$ are smaller than $y_k^{i_k}$.

The actual selection process defined in (b) above will be dictated by the branch and bound rule already specified. Note that, by construction,

$$T_0 \subset T_1 \subset \dots \subset T_p.$$

Note also that any particular vertex v^t of G has exactly $n+1$ neighbors, but not all such neighbors are eligible for appendage to T_k in defining T_{k+1} (except in moving from T_0 to T_1 , when *all* neighbors of $v^0 = (x^0, y^0)$ join v^0 to define T_1). In particular, terminal vertices v^t of the form $v^t = (x^t, 0)$ have *no* neighbors that are eligible candidates and hence cannot be used to proceed from T_k to T_{k+1} .

Associated with each subtree T_k will be a polytope C_k enclosing C , and hence a polytope S_k enclosing S . We now proceed to define these polytopes. Let $v^t = (x^t, y^t)$ be a vertex of T_k . Let $N(v^t)$ denote the set of all vertices $v^{t, i}$ which are neighbors of v^t

on C , and let $K(v^t)$ denote the cone in R^{n+1} whose vertex is v^t and whose defining rays are $r^{t,i} = \{v: v = v^t + \lambda(v^{t,i} - v^t); \lambda \geq 0\}$ for each $v^{t,i} \in N(v^t)$. The set C_k is defined to be

$$C_k = \bigcap_{v^t \in T_k} K(v^t)$$

i.e., C_k is the intersection of all cones defined at vertices of T_k .

We are primarily interested in the projection of the set C_k onto the hyperplane $y = 0$ which contains S . Thus we define

$$S_k = \{x: (x, y) \in C_k\}.$$

We now wish to show that

$$S_0 \supset S_1 \supset \dots \supset S_k \supset S. \quad (1)$$

Theorem 1: $S \subset S_0$.

Proof: C_0 is defined from T_0 , whose only vertex is $v^0 = (x^0, y^0)$, the point which solves problem CP. Thus C_0 is the cone defined by the $n+1$ inequalities

$$A_i x + \|A_i\| y \leq b_i$$

where i is such that $A_i x^0 + \|A_i\| y^0 = b_i$, since it has already been shown in Section 2 that there must exist precisely $n+1$ nonbasic slack variables in the tableau defining (x^0, y^0) . Now $\hat{x} \in S$ if and only if $A_i x \leq b_i$ for all $i=1, \dots, m$, so that $(\hat{x}, 0)$ must satisfy the above $n+1 \leq m$ inequalities.

Theorem 2: $S_{k+1} \subset S_k$.

Proof: As T_k is a subset of T_{k+1} , it follows that $C_{k+1} \subset C_k$. Hence $S_{k+1} = C_{k+1} \cap \{y=0\} \subset C_k \cap \{y=0\} = S_k$.

Theorem 3: $S \subset S_k$ for any k .

Proof: Any of the cones $K(v^t)$ defining C_k has the hyperplane description

$$K(v^t) = \left\{ (x,y): A_i x + ||A_i|| y \leq b_i \text{ for those } n+1 \text{ indices } i \text{ such that } A_i x^t + ||A_i|| y^t = b_i \right\}.$$

As such, it must contain the set S . Hence

$$S \subset K(v^t) \text{ for all vertices } v^t \text{ of } T_k,$$

so that

$$S \subset \bigcap_{v^t \in T_k} K(v^t).$$

These three theorems establish (1). We note in particular that any sequence of trees $T_0 \subset T_1 \subset T_2 \subset \dots$ must be finite. Indeed, the largest and last tree T_k generated in any such sequence must have all of the vertices of S among the terminal vertices of T_k . Mattheiss [2] actually generates such a tree T_k in order to list all vertices of S . He uses the distance measuring variable y^t of the terminal vertices $v^t = (x^t, y^t)$ of T_k to define T_{k+1} , and hence has control of the size and order of the list of terminal vertices of T_k .

Since we are interested in minimizing concave functions over S as opposed to generating all vertices of S , we will be using the value of the objective function f over the vertices of S_k to determine

T_{k+1} from T_k . We do this, of course, to avoid, whenever possible, the complete generation of all vertices of S . We will, however, use the values y^t to order our list of vertices of T_k for easy access.

In order to proceed from T_k to T_{k+1} , we need a *vertex description* of S_k (as opposed to the current description $S_k = C_k \cap \{y=0\}$). Consider any particular cone $K(v^t)$ defining C_k . Let $r^{t,i} = \{v: v = v^t + \lambda(v^{t,i} - v^t); \lambda \geq 0\}$ be one of the defining rays of C_k , where $v^{t,i}$ is a neighbor of v^t . Either $v^{t,i} \in T_k$, in which case $K(v^{t,i})$ is also a cone defining C_k , or $v^{t,i} \notin T_k$.

If $v^{t,i} \in T_k$, and if v^t is such that $y^t \leq y^{t,i}$, then the ray $r^{t,i}$ does not pierce the plane $y = 0$. For any point $v \in r^{t,i}$ has the form

$$v = (x, y) = (x^t + \lambda(x^{t,i} - x^t), y^t + \lambda(y^{t,i} - y^t))$$

so that $y > 0$ for all $\lambda \geq 0$.

If $v^{t,i} \in T_k$, and if we have $y^t > y^{t,i}$, then the ray $r^{t,i}$ does pierce the plane $y = 0$, but at a point $(e^{t,i}, 0)$, which lies outside of S_k . This follows since the ray $r^{t,i}$ is determined by precisely n hyperplanes (the n hyperplanes which v^t and $v^{t,i}$ have in common), and the point $v^{t,i}$ is that point where $r^{t,i}$ pierces the $(n+1)$ st hyperplane defining $v^{t,i}$ (see Figure 6). Since $r^{t,i}$ is feasible to C_k between v^t and $v^{t,i}$, it must become infeasible thereafter.

Finally, we must consider the case where $v^{t,i} \notin T_k$. Again, we have two subcases: $y^t \leq y^{t,i}$ and $y^t > y^{t,i}$. As before, in the first

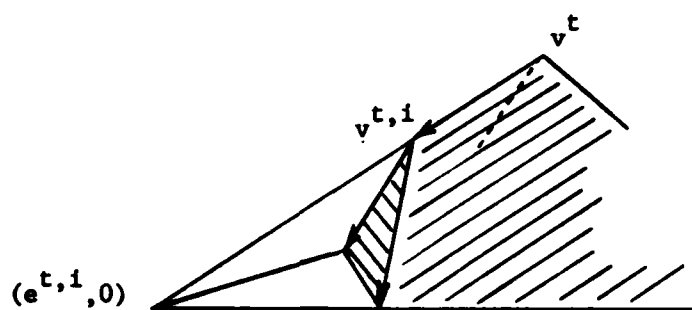


Figure 6.--An infeasible extension.

case, the ray $r^{t,i}$ does not pierce $y = 0$. In the second case, however, $r^{t,i}$ does pierce $y = 0$ in a point $(e^{t,i}, 0)$. Here, finally, we have a defining vertex of S_k . If, in fact, the point were *not* feasible to S_k , it would follow that the ray $r^{t,i}$ would have to pierce some other hyperplane defining C_k , and hence would have to pass through some neighbor of v^t . But we had assumed the contrary.

Summarizing the preceding discussion, we can define the generators of S_k to be those points $e^{t,i}$ found by selecting a *terminal vertex* v^t of T_k , and a neighbor $v^{t,i}$ of v^t such that

- (a) $v^{t,i}$ is not in T_k , and
- (b) $y^{t,i} < y^t$.

Then $e^{t,i} = x^t + \lambda_{t,i}(x^{t,i} - x^t)$ where

$$\lambda_{t,i} = \left(\frac{y^t}{y^t - y^{t,i}} \right) > 0. \quad (2)$$

The actual mechanics of computing the generators of S_k are not difficult. If we have the tableau of the full system $Ax + ay = b$ in a

form which exhibits $v^t = (x^t, y^t)$, we can easily determine which non-basic columns decrease the value of y when pivoted into the basics. The precise value of $\lambda_{t,i}$ above is then determined by driving y to 0 (ignoring feasibility).

Referring to Figure 7, suppose that we have the tableau representing $v^t = (x^t, y^t)$. Since $y^t > 0$, the variable y must be basic, and hence y must have the representation

$$y + \sum_j \tilde{a}_{1j} s_j^N = b_1.$$

Now a pivot in column j (i.e., make s_j^N basic) will result in a decreased value of y only if $\tilde{a}_{1j} > 0$. For such entries \tilde{a}_{1j} , a pivot in these entries themselves will decrease y to 0 (i.e., make y non-basic), although it may yield points x which are infeasible to S . The collection of all such x 's resulting in such pivots yields the new vertices of S_{k+1} , provided $v^{t,i}$ is not in T_k . The latter proviso requires a list search (the list of vertices of T_k), but is facilitated by the distance measure $y^{t,i}$.

x	y	s^B	s^N	rhs
	I			

Figure 7.--Tableau representing v^t .

4. Implementation and Computational Experience

As already stated, there are three phases to the collapsing polytopes algorithm: (1) the initialization phase, which requires the solution of a linear program and the subsequent generation of the enclosing polyhedron using simple pivot calculations; (2) the branching stage, which requires the removal from a list of the terminal node whose objective function value is minimal; and (3) the generation of new neighbors. This third stage requires that simplex-like pivots be performed and all nodes generated and not previously on the list be added to the list.

Our implementation of this algorithm uses the linear programming package SEXOP, developed in 1972 by Dr. Roy E. Marsten for all pivot calculations.* Two modifications have been made to SEXOP: (1) pivots can be made on any element in a given basis tableau, regardless of feasibility considerations; and (2) one can obtain the results of a pivot operation without actually changing the current tableau representation (we shall refer to such operations as pseudo-pivots).

In order to obtain the information necessary for branching strategies, a simple pointer-structured list is maintained which is ordered by objective function value. This list is linked to a second list ordered by the sum of the indices of the nonbasis variables which represent that vertex. This second list carries along with each nonbasis sum value, the indices of the nonbasic variables. This index information is all that is necessary to set up the tableau for any vertex.

*The authors are aware that Dr. Marsten has recently developed a new linear programming package, XMP, which is both more efficient and more stable than the earlier SEXOP package. XMP will replace SEXOP for all linear programming calculations in the concave minimization code as soon as the necessary modifications to that package have been made. However, since all computational results reported in this paper use SEXOP and since only counts on the number of pivot operations (rather than CPU time required for such pivots) are reported, we believe the comparative conclusions drawn will not be altered by the substitution.

At the branching stage, one need only remove the top entry from the first list and its associated index information from the second list. Given a branching node, one then generates its neighbors. One determines if each of these neighbors is added to the list by checking whether there exists a variable on the second list whose nonbasic indices sum equals that obtained for this new vertex. If such an entry exists, then one does a further check to insure that each nonbasic index matches. If it does, the vertex is not entered on the list. If a match does not occur, one also checks a "tombstone" list (also ordered by nonbasic indices sum) to assure that the vertex had not been generated previously and subsequently removed. If no match occurs on either list, this new vertex is entered in both the list ordered by objective function value and linked to the list ordered by nonbasic indices sum. We shall refer to the computer implementation of this algorithm as CONCAVE.

To evaluate how well this code performs on a variety of problem types, it was compared to two other computer codes. The first, SUM, is an implementation of the earlier algorithm developed by the authors. This code is modularly designed with all input, output, simplex operations, and (where possible) list structure routines identical to those used by CONCAVE.

As mentioned earlier, SUM requires that both a tableau and a constraint representation of each vertex on the list be maintained. The other features of this algorithm which differ in code structure from that of CONCAVE are:

- (1) A linear program is solved in order to find the leaving vertex (branch selection).
- (2) Once the leaving vertex is chosen, one must remove it from the list of vertices which together determine the enclosing polytope. Since this vertex is not necessarily the vertex with the lowest objective function value, a search of the list for this specific vertex must take place. Having

ordered the list in the same manner as that for CONCAVE does, however, help in the search since the vertex sought is the one which contributed most to the low value of the underestimation function, i.e., an infeasible vertex with small objective function value.

- (3) After determining the leaving vertex, one then finds the associated constraint to be added. One must then be certain that no other vertices on the list violate this constraint. (If any does, it too must be removed.) Thus a test of each (infeasible) vertex must be made.
- (4) This algorithm requires that only one list be maintained, ordered by objective function value, with the values of the vertices and their basis representation carried along. No "tombstone" list need be maintained by CONCAVE.

The third code which was used in the comparison is a code developed by T. Mattheiss, called ALVERT, which generates all vertices of a polytope. Obviously, one method of solving concave minimization problems is to generate all vertices and choose the one whose objective function is minimum. The code ALVERT was modified to store not all vertices generated, but rather only those needed for future calculations (nonterminal vertices) and the one terminal vertex whose objective function value is minimal among all terminal vertices generated so far.

The Testing

All computational testing was performed on a UNIVAC 1108 computer under EXEC level 36R2D Operating System, using the FTN Compiler, level 9R1. This FORTRAN compiler is an optimizing compiler allowing many of the new syntax rules of the ANSI FORTRAN 77 standard.

We used as our performance measures: (1) the number of pseudo-pivots, (2) the number of actual pivots, and (3) the maximum number of elements on the list. We do not report any accuracy measures since all

three codes produced the same answer both in objective function value and in solution vector (to 10^{-5} accuracy) on all problems tested.

We again ran the four problems presented in [1]. The results appear in Table 1. In addition to these small examples used primarily for feasibility testing, the major test effort involved the pseudorandom generation of polytopes coupled with the generation of concave objective functions having linear fixed charges or negative quadratic terms. For each problem type, 25 problems were randomly generated and the numbers reported are the averages obtained from those 25 problems. In the tables below we present the results of this test effort. Note that for problems of a specific size, all that was altered between tables was the type of objective function to be minimized. ALVERT was, therefore, run only once for each problem size and this run summarized the results of generating all vertices for each of the 25 problems of that type.

A number of conclusions can be drawn from the data obtained in this test effort:

- (1) For problems of the size tested, neither the size of the fixed charge nor the size of the negative quadratic term has a significant effect on the computational effort required to solve these problems.
- (2) Both SUM and CONCAVE appear to outperform ALVERT for the types of problems tested.
- (3) CONCAVE appears to be superior to SUM for problems of the types tested.

These results indicate that successively underestimating the objective function by enclosing the feasible region in polytopes which decrease in size at each iteration is a promising approach for concave minimization problems.

TABLE 1
EXAMPLE PROBLEMS

Types of Concave Functions	Size of a Matrix	CONCAVE			SUM			ALVERT		
		# of Pivots	# of Actual Pivots	Max. List Size	# of Pseudo Pivots	# of Actual Pivots	Max. List Size	# of Pseudo Pivots	# of Actual Pivots	Max. List Size
Linear Fixed Charge Problem	7×4	5	5	1	10	12	8	6	4	1
Nonlinear Fixed Charge Problem	7×3	4	5	1	6	9	5	5	4	1
Negative Quadratic Problem	14×6	14	10	3	41	16	36	25	20	14
Grotte's Weapon Allocation Problem $\exp(-\sum a_i x_i)$	12×9	26	22	6	83	20	45	84	38	22

TABLE 2

LINEAR FIXED CHARGE PROBLEMS

5 × 10; Excluding Nonnegativities and Slack Variables

	Small Fixed Charge (same size as linear costs)			Medium Fixed Charge (twice size of linear costs)			Large Fixed Charge (5 times size of linear costs)		
	# of Pseudo Pivots	# of True List Size	Max. Size	# of Pseudo Pivots	# of True List Size	Max. Size	# of Pseudo Pivots	# of True List Size	Max. Size
CONCAVE	16.8	15	7.2	14.3	15	7.0	14.2	15.3	3.0
SUM	20.8	56.2	11.1	10.4	56.4	11.1	10.3	57.2	11.3
ALVERT	74.8	33.2	39.2	74.8	33.2	39.2	74.8	33.2	39.2

TABLE 3

LINEAR FIXED CHARGE PROBLEMS

10 × 20; Excluding Nonnegativities and Slack Variables

	Small Fixed Charge (same size as linear costs)			Medium Fixed Charge (twice size of linear costs)			Large Fixed Charge (5 times size of linear costs)		
	# of Pseudo Pivots	# of True List Size	Max. Size	# of Pseudo Pivots	# of True List Size	Max. Size	# of Pseudo Pivots	# of True List Size	Max. Size
CONCAVE	21.8	30.4	5.0	24.8	31.3	5.3	10.4	33.7	5.4
SUM	20.4	63.2	11.0	12.4	64.5	10.9	14.3	57.1	11.4
ALVERT	394.6	171.3	92.8	394.6	171.3	92.8	394.6	171.3	92.8

TABLE 4

NEGATIVE QUADRATIC PROBLEMS; 5×10

	Small Quadratic Cost (same size as linear costs)			Medium Quadratic Cost (twice size of linear costs)			Large Quadratic Cost (5 times size of linear costs)		
	# of Pseudo Pivots	# of True Pivots	Max. List Size	# of Pseudo Pivots	# of True Pivots	Max. List Size	# of Pseudo Pivots	# of True Pivots	Max. List Size
CONCAVE	55.2	12.4	10.0	48.7	18.6	11.1	64.3	16.3	9.4
SUM	25.8	60.8	15.0	36.4	73.4	28.2	32.7	65.3	35.6
ALVERT	74.8	33.2	39.2	74.8	33.2	39.2	74.8	33.2	39.2

TABLE 5

NEGATIVE QUADRATIC PROBLEMS; 10×20

	Small Quadratic Cost (same size as linear costs)			Medium Quadratic Cost (twice size of linear costs)			Large Quadratic Cost (5 times size of linear costs)		
	# of Pseudo Pivots	# of True Pivots	Max. List Size	# of Pseudo Pivots	# of True Pivots	Max. List Size	# of Pseudo Pivots	# of True Pivots	Max. List Size
CONCAVE	16.8	15.4	7.8	19.8	14.3	5.6	22.4	11.6	9.4
SUM	22.4	104.7	32.0	26.8	103.9	41.7	31.3	120.3	36.8
ALVERT	394.6	171.3	92.8	394.6	171.3	92.8	394.6	171.3	92.8

REFERENCES

- [1] Falk, J. E. and K. R. Hoffman (1976). A successive underestimation method for concave minimization problems, *Mathematics of Operations Research*, Vol. 1, No. 3, pp. 251-259.
- [2] Mattheiss, T. H. (1973). An algorithm for the determination of irrelevant constraints in all vertices in systems of linear inequalities, *Operations Research*, Vol. XXI, pp. 247-260.
- [3] Zwart, P. B. (1973). Nonlinear programming: Counterexamples to two global optimization algorithms, *Operations Research*, Vol. 1, pp. 1260-1266.
- [4] Tui, H. (1964). Concave programming under linear constraints, *Doklady Akademii Nauk SSSR*, Vol. 159, pp. 32-35. Translated (1964), in *Soviet Mathematics, Doklady*, Vol. 4, pp. 1437-1440.

THE GEORGE WASHINGTON UNIVERSITY
Program in Logistics
Distribution List for Technical Papers

The George Washington University Office of Sponsored Research Gelman Library Vice President H. F. Bright Dean Harold Liebowitz Dean Henry Solomon	Armed Forces Industrial College Armed Forces Staff College Army War College Library Carlisle Barracks Army Cmd & Gen Staff College Army Logistics Mgt Center Fort Lee Commanding Officer, USALDSRA New Cumberland Army Depot Army Inventory Res Ofc Philadelphia Army Trans Material Cmd TCMAC-ASDT Air Force Headquarters AFADS-3 LEXY SAF/ALG Griffiss Air Force Base Reliability Analysis Center Gunter Air Force Base AFLMC/XR Maxwell Air Force Base Library Wright-Patterson Air Force Base AFLC/OA Research Sch Log AFALD/XR Defense Technical Info Center National Academy of Sciences Maritime Transportation Res Bd Lib National Bureau of Standards Dr B. H. Colvin Dr Joan Rosenblatt National Science Foundation National Security Agency Weapons Systems Evaluation Group British Navy Staff National Defense Hdqtrs, Ottawa Logistics, OR Analysis Estab American Power Jet Co George Chernowitz General Dynamics, Pomona General Research Corp Library Mr William P. Hutzler Logistics Management Institute Dr Murray A. Geisler Rand Corporation Library Mr William P. Hutzler Carnegie-Mellon University Dean H. A. Simon Prof G. Thompson	Case Western Reserve University Prof B. V. Dean Prof M. Mesarovic Cornell University Prof R. E. Bechhofer Prof R. W. Conway Prof Andrew Schultz, Jr. Cowles Foundation for Research in Economics Prof Martin Shubik Florida State University Prof R. A. Bradley Harvard University Prof W. G. Cochran Prof Arthur Schleifer, Jr. Princeton University Prof A. W. Tucker Prof J. W. Tukey Prof Geoffrey S. Watson Purdue University Prof S. S. Gupta Prof H. Rubin Prof Andrew Whinston Stanford University Prof T. W. Anderson Prof Kenneth Arrow Prof G. B. Dantzig Prof F. S. Hillier Prof D. L. Iglehart Prof Samuel Karlin Prof G. J. Lieberman Prof Herbert Solomon Prof A. F. Veinott, Jr. University of California, Berkeley Prof R. E. Barlow Prof D. Gale Prof Jack Kiefer University of California, Los Angeles Prof R. R. O'Neill University of North Carolina Prof W. L. Smith Prof M. R. Leadbetter University of Pennsylvania Prof Russell Ackoff University of Texas Institute for Computing Science and Computer Applications Yale University Prof F. J. Anscombe Prof H. Scarf Prof Z. W. Birnbaum University of Washington Prof B. H. Bissinger The Pennsylvania State University Prof Seth Bonder University of Michigan Prof G. E. Box University of Wisconsin Dr Jerome Bracken Institute for Defense Analyses
ONR Chief of Naval Research (Codes 200, 434) Resident Representative OPNAV OP-40 DCNO, Logistics Navy Dept Library NAVDATA Automation Cmd Naval Aviation Integrated Log Suppo NARDAC Tech Library Naval Electronics Lab Library Naval Facilities Eng Cmd Tech Library Naval Ordnance Station Louisville, Ky. Indian Head, Md. Naval Ordnance Sys Cmd Library Naval Research Branch Office Boston Chicago New York Pasadena San Francisco Naval Ship Eng Center Philadelphia, Pa. Naval Ship Res & Dev Center Naval Sea Systems Command PMS 30611 Tech Library Code 073 Naval Supply Systems Command Library Operations and Inventory Analysis Naval War College Library Newport BUPERS Tech Library FMSO USN Ammo Depot Earle USN Postgrad School Monterey Library Dr Jack R. Borsting Prof C. R. Jones US Coast Guard Academy Capt Jimmie D. Woods US Marine Corps Commandant Deputy Chief of Staff, R&D Marine Corps School Quantico Landing Force Dev Ctr Logistics Officer		

Continued

Prof A. Charnes
University of Texas

Prof H. Chernoff
Mass Institute of Technology

Prof Arthur Cohen
Rutgers - The State University

Mr Wallace M. Cohen
US General Accounting Office

Prof C. Derman
Columbia University

Prof Masao Fukushima
Kyoto University

Prof Saul I. Gass
University of Maryland

Dr Donald P. Gaver
Carmel, California

Prof Amrit L. Goel
Syracuse University

Prof J. F. Hannan
Michigan State University

Prof H. O. Hartley
Texas A & M Foundation

Prof W. M. Hirsch
Courant Institute

Dr Alan J. Hoffman
IBM, Yorktown Heights

Prof John R. Isbell
SUNY, Amherst

Dr J. L. Jain
University of Delhi

Prof J. H. K. Kao
Polytech Institute of New York

Prof W. Kruskal
University of Chicago

Mr S. Kumar
University of Madras

Prof C. E. Lemke
Rensselaer Polytech Institute

Prof Loynes
University of Sheffield, England

Prof Tom Maul
Kowloon, Hong Kong

Prof Steven Nahmias
University of Santa Clara

Prof D. B. Owen
Southern Methodist University

Prof P. R. Parathasarathy
Indian Institute of Technology

Prof E. Parzen
Texas A & M University

Prof H. O. Posten
University of Connecticut

Prof R. Remage, Jr.
University of Delaware

Prof Hans Riedwyl
University of Berne

Mr David Rosenblatt
Washington, D. C.

Prof M. Rosenblatt
University of California, San Diego

Prof Alan J. Rowe
University of Southern California

Prof A. H. Rubenstein
Northwestern University

Prof Thomas L. Saaty
University of Pittsburgh

Dr M. E. Salvesson
West Los Angeles

Prof Gary Scudder
University of Minnesota

Prof Edward A. Silver
University of Waterloo, Canada

Prof Rosedith Sitgreaves
Washington, DC

LTC G. L. Slyman, MSC
Department of the Army

Prof M. J. Sobel
Georgia Inst of Technology

Prof R. M. Thrall
Rice University

Dr S. Vajda
University of Sussex, England

Prof T. M. Whitin
Wesleyan University

Prof Jacob Wolfowitz
University of South Florida

Prof Max A. Woodbury
Duke University

Prof S. Zacks
SUNY, Binghamton

Dr Israel Zang
Tel-Aviv University